Summary
15/01/2017, 11:33:46

Differences exist between documents.

New Document:	Old Document:
Configuration_adv	Configuration_adv or
26 pages (108 KB)	26 pages (108 KB)
15/01/2017, 11:33:41	15/01/2017, 11:33:41
Used to display results.	

Get started: first change is on page 3.

No pages were deleted

How to read this report

Highlight indicates a change.
~~**Deleted**~~ indicates deleted content.
 indicates pages were changed.
 indicates pages were moved.

```
/**
 * Marlin 3D Printer Firmware
 * Copyright (C) 2016 MarlinFirmware [https://github.com/
MarlinFirmware/Marlin]
 *
 * Based on Sprinter and grbl.
 * Copyright (C) 2011 Camiel Gubbels / Erik van der Zalm
 *
 * This program is free software: you can redistribute it and/or
modify
 * it under the terms of the GNU General Public License as published
by
 * the Free Software Foundation, either version 3 of the License, or
 * (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program.  If not, see <http://www.gnu.org/
licenses/>.
 *
 */

/**
 * Configuration_adv.h
 *
 * Advanced settings.
 * Only change these if you know exactly what you're doing.
 * Some of these settings can damage your printer if improperly set!
 *
 * Basic settings can be found in Configuration.h
 *
 */
#ifndef CONFIGURATION_ADV_H
#define CONFIGURATION_ADV_H

/**
 *
 *  ***********************************
 *  **  ATTENTION TO ALL DEVELOPERS  **
 *  ***********************************
 *
 * You must increment this version number for every significant
change such as,
 * but not limited to: ADD, DELETE RENAME OR REPURPOSE any
directive/option.
 *
 * Note: Update also Version.h !
 */
#define CONFIGURATION_ADV_H_VERSION 010100
```

```
// @section temperature

//
===========================================================================
=======
//=============================Thermal Settings
============================
//
===========================================================================
=======

#if DISABLED(PIDTEMPBED)
  #define BED_CHECK_INTERVAL 5000 // ms between checks in bang-bang
control
  #if ENABLED(BED_LIMIT_SWITCHING)
    #define BED_HYSTERESIS 2 // Only disable heating if
T>target+BED_HYSTERESIS and enable heating if T>target-
BED_HYSTERESIS
  #endif
#endif

/**
 * Thermal Protection protects your printer from damage and fire if
a
 * thermistor falls out or temperature sensors fail in any way.
 *
 * The issue: If a thermistor falls out or a temperature sensor
fails,
 * Marlin can no longer sense the actual temperature. Since a
disconnected
 * thermistor reads as a low temperature, the firmware will keep the
heater on.
 *
 * The solution: Once the temperature reaches the target, start
observing.
 * If the temperature stays too far below the target (hysteresis)
for too long (period),
 * the firmware will halt the machine as a safety precaution.
 *
 * If you get false positives for "Thermal Runaway" increase
THERMAL_PROTECTION_HYSTERESIS and/or THERMAL_PROTECTION_PERIOD
 */
#if ENABLED(THERMAL_PROTECTION_HOTENDS)
  #define THERMAL_PROTECTION_PERIOD 40        // Seconds
  #define THERMAL_PROTECTION_HYSTERESIS 4     // Degrees Celsius

  /**
   * Whenever an M104 or M109 increases the target temperature the
firmware will wait for the
   * WATCH_TEMP_PERIOD to expire, and if the temperature hasn't
increased by WATCH_TEMP_INCREASE
   * degrees, the machine is halted, requiring a hard reset. This
test restarts with any M104/M109,
   * but only if the current temperature is far enough below the
```

```
target for a reliable test.
   *
   * If you get false positives for "Heating failed" increase
WATCH_TEMP_PERIOD and/or decrease WATCH_TEMP_INCREASE
   * WATCH_TEMP_INCREASE should not be below 2.
   */
  #define WATCH_TEMP_PERIOD 30             // Seconds
  #define WATCH_TEMP_INCREASE 2            // Degrees Celsius
#endif

/**
 * Thermal Protection parameters for the bed are just as above for
hotends.
 */
#if ENABLED(THERMAL_PROTECTION_BED)
  #define THERMAL_PROTECTION_BED_PERIOD 20    // Seconds
  #define THERMAL_PROTECTION_BED_HYSTERESIS 2 // Degrees Celsius

  /**
   * Whenever an M140 or M190 increases the target temperature the
firmware will wait for the
   * WATCH_BED_TEMP_PERIOD to expire, and if the temperature hasn't
increased by WATCH_BED_TEMP_INCREASE
   * degrees, the machine is halted, requiring a hard reset. This
test restarts with any M140/M190,
   * but only if the current temperature is far enough below the
target for a reliable test.
   *
   * If you get too many "Heating failed" errors, increase
WATCH_BED_TEMP_PERIOD and/or decrease
   * WATCH_BED_TEMP_INCREASE. (WATCH_BED_TEMP_INCREASE should not be
below 2.)
   */
  #define WATCH_BED_TEMP_PERIOD 90             // Seconds
  #define WATCH_BED_TEMP_INCREASE 2            // Degrees Celsius
#endif

#if ENABLED(PIDTEMP)
  // this adds an experimental additional term to the heating power,
proportional to the extrusion speed.
  // if Kc is chosen well, the additional required power due to
increased melting should be compensated.
  //#define PID_EXTRUSION_SCALING
  #if ENABLED(PID_EXTRUSION_SCALING)
    #define DEFAULT_Kc (100) //heating power=Kc*(e_speed)
    #define LPQ_MAX_LEN 50
  #endif
#endif

/**
 * Automatic Temperature:
 * The hotend target temperature is calculated by all the buffered
lines of gcode.
 * The maximum buffered steps/sec of the extruder motor is called
```

```
"se".
 * Start autotemp mode with M109 S<mintemp> B<maxtemp> F<factor>
 * The target temperature is set to mintemp+factor*se[steps/sec] and
is limited by
 * mintemp and maxtemp. Turn this off by executing M109 without F*
 * Also, if the temperature is set to a value below mintemp, it will
not be changed by autotemp.
 * On an Ultimaker, some initial testing worked with M109 S215 B260
F1 in the start.gcode
 */
#define AUTOTEMP
#if ENABLED(AUTOTEMP)
  #define AUTOTEMP_OLDWEIGHT 0.98
#endif

//Show Temperature ADC value
//The M105 command return, besides traditional information, the ADC
value read from temperature sensors.
//#define SHOW_TEMP_ADC_VALUES

/**
 * High Temperature Thermistor Support
 *
 * Thermistors able to support high temperature tend to have a hard
time getting
 * good readings at room and lower temperatures. This means
HEATER_X_RAW_LO_TEMP
 * will probably be caught when the heating element first turns on
during the
 * preheating process, which will trigger a min_temp_error as a
safety measure
 * and force stop everything.
 * To circumvent this limitation, we allow for a preheat time
(during which,
 * min_temp_error won't be triggered) and add a min_temp buffer to
handle
 * aberrant readings.
 *
 * If you want to enable this feature for your hotend thermistor(s)
 * uncomment and set values > 0 in the constants below
 */

// The number of consecutive low temperature errors that can occur
// before a min_temp_error is triggered. (Shouldn't be more than
10.)
//#define MAX_CONSECUTIVE_LOW_TEMPERATURE_ERROR_ALLOWED 0

// The number of milliseconds a hotend will preheat before starting
to check
// the temperature. This value should NOT be set to the time it
takes the
// hot end to reach the target temperature, but the time it takes to
reach
// the minimum temperature your thermistor can read. The lower the
```

```
better/safer.
// This shouldn't need to be more than 30 seconds (30000)
//#define MILLISECONDS_PREHEAT_TIME 0

// @section extruder

// Extruder runout prevention.
// If the machine is idle and the temperature over MINTEMP
// then extrude some filament every couple of SECONDS.
//#define EXTRUDER_RUNOUT_PREVENT
#if ENABLED(EXTRUDER_RUNOUT_PREVENT)
  #define EXTRUDER_RUNOUT_MINTEMP 190
  #define EXTRUDER_RUNOUT_SECONDS 30
  #define EXTRUDER_RUNOUT_SPEED 1500  // mm/m
  #define EXTRUDER_RUNOUT_EXTRUDE 5   // mm
#endif

// @section temperature

//These defines help to calibrate the AD595 sensor in case you get
wrong temperature measurements.
//The measured temperature is defined as "actualTemp = (measuredTemp
* TEMP_SENSOR_AD595_GAIN) + TEMP_SENSOR_AD595_OFFSET"
#define TEMP_SENSOR_AD595_OFFSET 0.0
#define TEMP_SENSOR_AD595_GAIN   1.0

//This is for controlling a fan to cool down the stepper drivers
//it will turn on when any driver is enabled
//and turn off after the set amount of seconds from last driver
being disabled again
#define CONTROLLERFAN_PIN 11 //Pin used for the fan to cool
controller (-1 to disable)
#define CONTROLLERFAN_SECS 60 //How many seconds, after all motors
were disabled, the fan should run
#define CONTROLLERFAN_SPEED 255  // == full speed

// When first starting the main fan, run it at full speed for the
// given number of milliseconds.  This gets the fan spinning
reliably
// before setting a PWM value. (Does not work with software PWM for
fan on Sanguinololu)
#define FAN_KICKSTART_TIME 100

// This defines the minimal speed for the main fan, run in PWM mode
// to enable uncomment and set minimal PWM speed for reliable
running (1-255)
// if fan speed is [1 - (FAN_MIN_PWM-1)] it is set to FAN_MIN_PWM
//#define FAN_MIN_PWM 50

// @section extruder

/**
 * Extruder cooling fans
 *
```

```
 * Extruder auto fans automatically turn on when their extruders'
 * temperatures go above EXTRUDER_AUTO_FAN_TEMPERATURE.
 *
 * Your board's pins file specifies the recommended pins. Override
those here
 * or set to -1 to disable completely.
 *
 * Multiple extruders can be assigned to the same pin in which case
 * the fan will turn on when any selected extruder is above the
threshold.
 */
#define E0_AUTO_FAN_PIN 9
#define E1_AUTO_FAN_PIN -1
#define E2_AUTO_FAN_PIN -1
#define E3_AUTO_FAN_PIN -1
#define EXTRUDER_AUTO_FAN_TEMPERATURE 50
#define EXTRUDER_AUTO_FAN_SPEED   255  // == full speed

// Define a pin to turn case light on/off
//#define CASE_LIGHT_PIN 4
#if PIN_EXISTS(CASE_LIGHT)
  #define INVERT_CASE_LIGHT false   // Set to true if HIGH is the
OFF state (active low)
  //#define CASE_LIGHT_DEFAULT_ON   // Uncomment to set default
state to on
  //#define MENU_ITEM_CASE_LIGHT    // Uncomment to have a Case
Light On / Off entry in main menu
#endif

//
================================================================
=======
//============================ Mechanical Settings
=========================
//
================================================================
=======

// @section homing

// If you want endstops to stay on (by default) even when not homing
// enable this option. Override at any time with M120, M121.
//#define ENDSTOPS_ALWAYS_ON_DEFAULT

// @section extras

//#define Z_LATE_ENABLE // Enable Z the last moment. Needed if your
Z driver overheats.

// Dual X Steppers
// Uncomment this option to drive two X axis motors.
// The next unused E driver will be assigned to the second X
stepper.
//#define X_DUAL_STEPPER_DRIVERS
```

```
#if ENABLED(X_DUAL_STEPPER_DRIVERS)
  // Set true if the two X motors need to rotate in opposite
directions
  #define INVERT_X2_VS_X_DIR true
#endif


// Dual Y Steppers
// Uncomment this option to drive two Y axis motors.
// The next unused E driver will be assigned to the second Y
stepper.
//#define Y_DUAL_STEPPER_DRIVERS
#if ENABLED(Y_DUAL_STEPPER_DRIVERS)
  // Set true if the two Y motors need to rotate in opposite
directions
  #define INVERT_Y2_VS_Y_DIR true
#endif

// A single Z stepper driver is usually used to drive 2 stepper
motors.
// Uncomment this option to use a separate stepper driver for each Z
axis motor.
// The next unused E driver will be assigned to the second Z
stepper.
#define Z_DUAL_STEPPER_DRIVERS

#if ENABLED(Z_DUAL_STEPPER_DRIVERS)

  // Z_DUAL_ENDSTOPS is a feature to enable the use of 2 endstops
for both Z steppers - Let's call them Z stepper and Z2 stepper.
  // That way the machine is capable to align the bed during home,
since both Z steppers are homed.
  // There is also an implementation of M666 (software endstops
adjustment) to this feature.
  // After Z homing, this adjustment is applied to just one of the
steppers in order to align the bed.
  // One just need to home the Z axis and measure the distance
difference between both Z axis and apply the math: Z adjust = Z -
Z2.
  // If the Z stepper axis is closer to the bed, the measure Z > Z2
(yes, it is.. think about it) and the Z adjust would be positive.
  // Play a little bit with small adjustments (0.5mm) and check the
behaviour.
  // The M119 (endstops report) will start reporting the Z2 Endstop
as well.

  #define Z_DUAL_ENDSTOPS

  #if ENABLED(Z_DUAL_ENDSTOPS)
    #define Z2_USE_ENDSTOP _ZMAX_
  #endif

#endif // Z_DUAL_STEPPER_DRIVERS
```

```
// Enable this for dual x-carriage printers.
// A dual x-carriage design has the advantage that the inactive
extruder can be parked which
// prevents hot-end ooze contaminating the print. It also reduces
the weight of each x-carriage
// allowing faster printing speeds. Connect your X2 stepper to the
first unused E plug.
//#define DUAL_X_CARRIAGE
#if ENABLED(DUAL_X_CARRIAGE)
  // Configuration for second X-carriage
  // Note: the first x-carriage is defined as the x-carriage which
homes to the minimum endstop;
  // the second x-carriage always homes to the maximum endstop.
  #define X2_MIN_POS 80     // set minimum to ensure second x-
carriage doesn't hit the parked first X-carriage
  #define X2_MAX_POS 353    // set maximum to the distance between
toolheads when both heads are homed
  #define X2_HOME_DIR 1     // the second X-carriage always homes to
the maximum endstop position
  #define X2_HOME_POS X2_MAX_POS // default home position is the
maximum carriage position
      // However: In this mode the HOTEND_OFFSET_X value for the
second extruder provides a software
      // override for X2_HOME_POS. This also allow recalibration of
the distance between the two endstops
      // without modifying the firmware (through the "M218 T1 X???"
command).
      // Remember: you should set the second extruder x-offset to 0
in your slicer.

  // There are a few selectable movement modes for dual x-carriages
using M605 S<mode>
  //    Mode 0: Full control. The slicer has full control over both
x-carriages and can achieve optimal travel results
  //                            as long as it supports dual x-
carriages. (M605 S0)
  //    Mode 1: Auto-park mode. The firmware will automatically park
and unpark the x-carriages on tool changes so
  //                            that additional slicer support is not
required. (M605 S1)
  //    Mode 2: Duplication mode. The firmware will transparently
make the second x-carriage and extruder copy all
  //                            actions of the first x-carriage. This
allows the printer to print 2 arbitrary items at
  //                            once. (2nd extruder x offset and temp
offset are set using: M605 S2 [Xnnn] [Rmmm])

  // This is the default power-up mode which can be later using
M605.
  #define DEFAULT_DUAL_X_CARRIAGE_MODE DXC_FULL_CONTROL_MODE

  // Default settings in "Auto-park Mode"
  #define TOOLCHANGE_PARK_ZLIFT   0.2      // the distance to raise
Z axis when parking an extruder
```

```
  #define TOOLCHANGE_UNPARK_ZLIFT 1       // the distance to raise
Z axis when unparking an extruder

  // Default x offset in duplication mode (typically set to half
print bed width)
  #define DEFAULT_DUPLICATION_X_OFFSET 100

#endif //DUAL_X_CARRIAGE

// @section homing

//homing hits the endstop, then retracts by this distance, before it
tries to slowly bump again:
#define X_HOME_BUMP_MM 5
#define Y_HOME_BUMP_MM 5
#define Z_HOME_BUMP_MM 2
#define HOMING_BUMP_DIVISOR {2, 2, 4}  // Re-Bump Speed Divisor
(Divides the Homing Feedrate)
//#define QUICK_HOME  //if this is defined, if both x and y are to
be homed, a diagonal move will be performed initially.

// When G28 is called, this option will make Y home before X
//#define HOME_Y_BEFORE_X

// @section machine

#define AXIS_RELATIVE_MODES {false, false, false, false}

// Allow duplication mode with a basic dual-nozzle extruder
//#define DUAL_NOZZLE_DUPLICATION_MODE

// By default pololu step drivers require an active high signal.
However, some high power drivers require an active low signal as
step.
#define INVERT_X_STEP_PIN false
#define INVERT_Y_STEP_PIN false
#define INVERT_Z_STEP_PIN false
#define INVERT_E_STEP_PIN false

// Default stepper release if idle. Set to 0 to deactivate.
// Steppers will shut down DEFAULT_STEPPER_DEACTIVE_TIME seconds
after the last move when DISABLE_INACTIVE_? is true.
// Time can be set by M18 and M84.
#define DEFAULT_STEPPER_DEACTIVE_TIME 120
#define DISABLE_INACTIVE_X true
#define DISABLE_INACTIVE_Y true
#define DISABLE_INACTIVE_Z true  // set to false if the nozzle will
fall down on your printed part when print has finished.
#define DISABLE_INACTIVE_E true

#define DEFAULT_MINIMUMFEEDRATE       0.0     // minimum feedrate
#define DEFAULT_MINTRAVELFEEDRATE     0.0

// @section lcd
```

```
#if ENABLED(ULTIPANEL)
  #define MANUAL_FEEDRATE {50*60, 50*60, 5*60, 60} // Feedrates for
manual moves along X, Y, Z, E from panel
  #define ULTIPANEL_FEEDMULTIPLY  // Comment to disable setting
feedrate multiplier via encoder
#endif

// @section extras

// minimum time in microseconds that a movement needs to take if the
buffer is emptied.
#define DEFAULT_MINSEGMENTTIME        20000

// If defined the movements slow down when the look ahead buffer is
only half full
#define SLOWDOWN

// Frequency limit
// See nophead's blog for more info
// Not working O
//#define XY_FREQUENCY_LIMIT  15

// Minimum planner junction speed. Sets the default minimum speed
the planner plans for at the end
// of the buffer and all stops. This should not be much greater than
zero and should only be changed
// if unwanted behavior is observed on a user's machine when running
at very slow speeds.
#define MINIMUM_PLANNER_SPEED 0.05// (mm/sec)

// Microstep setting (Only functional when stepper driver microstep
pins are connected to MCU.
#define MICROSTEP_MODES {16,16,16,16,16} // [1,2,4,8,16]

// Motor Current setting (Only functional when motor driver current
ref pins are connected to a digital trimpot on supported boards)
#define DIGIPOT_MOTOR_CURRENT {135,135,135,135,135} // Values 0-255
(RAMBO 135 = ~0.75A, 185 = ~1A)

// Motor Current controlled via PWM (Overridable on supported boards
with PWM-driven motor driver current)
//#define PWM_MOTOR_CURRENT {1300, 1300, 1250} // Values in
milliamps

// uncomment to enable an I2C based DIGIPOT like on the Azteeg X3
Pro
//#define DIGIPOT_I2C
// Number of channels available for I2C digipot, For Azteeg X3 Pro
we have 8
#define DIGIPOT_I2C_NUM_CHANNELS 8
// actual motor currents in Amps, need as many here as
DIGIPOT_I2C_NUM_CHANNELS
#define DIGIPOT_I2C_MOTOR_CURRENTS {1.0, 1.0, 1.0, 1.0, 1.0, 1.0,
```

```
1.0, 1.0}

//
=============================================================================
=======
//============================Additional
Features============================
//
=============================================================================
=======

#define ENCODER_RATE_MULTIPLIER         // If defined, certain menu
edit operations automatically multiply the steps when the encoder is
moved quickly
#define ENCODER_10X_STEPS_PER_SEC 75    // If the encoder steps per
sec exceeds this value, multiply steps moved x10 to quickly advance
the value
#define ENCODER_100X_STEPS_PER_SEC 160  // If the encoder steps per
sec exceeds this value, multiply steps moved x100 to really quickly
advance the value

//#define CHDK 4        //Pin for triggering CHDK to take a picture
see how to use it here http://captain-slow.dk/2014/03/09/3d-
printing-timelapses/
#define CHDK_DELAY 50 //How long in ms the pin should stay HIGH
before going LOW again

// @section lcd

// Include a page of printer information in the LCD Main Menu
//#define LCD_INFO_MENU

// On the Info Screen, display XY with one decimal place when
possible
//#define LCD_DECIMAL_SMALL_XY

#if ENABLED(SDSUPPORT)

  // Some RAMPS and other boards don't detect when an SD card is
inserted. You can work
  // around this by connecting a push button or single throw switch
to the pin defined
  // as SD_DETECT_PIN in your board's pins definitions.
  // This setting should be disabled unless you are using a push
button, pulling the pin to ground.
  // Note: This is always disabled for ULTIPANEL (except
ELB_FULL_GRAPHIC_CONTROLLER).
  #define SD_DETECT_INVERTED

  #define SD_FINISHED_STEPPERRELEASE true  //if sd support and the
file is finished: disable steppers?
  #define SD_FINISHED_RELEASECOMMAND "M84 X Y Z E" // You might want
to keep the z enabled so your bed stays in place.
```

```
  #define SDCARD_RATHERRECENTFIRST  //reverse file order of sd card
menu display. Its sorted practically after the file system block
order.
  // if a file is deleted, it frees a block. hence, the order is not
purely chronological. To still have auto0.g accessible, there is
again the option to do that.
  // using:
  //#define MENU_ADDAUTOSTART

  // Show a progress bar on HD44780 LCDs for SD printing
  //#define LCD_PROGRESS_BAR

  #if ENABLED(LCD_PROGRESS_BAR)
    // Amount of time (ms) to show the bar
    #define PROGRESS_BAR_BAR_TIME 2000
    // Amount of time (ms) to show the status message
    #define PROGRESS_BAR_MSG_TIME 3000
    // Amount of time (ms) to retain the status message (0=forever)
    #define PROGRESS_MSG_EXPIRE   0
    // Enable this to show messages for MSG_TIME then hide them
    //#define PROGRESS_MSG_ONCE
  #endif

  // This allows hosts to request long names for files and folders
with M33
  //#define LONG_FILENAME_HOST_SUPPORT

  // This option allows you to abort SD printing when any endstop is
triggered.
  // This feature must be enabled with "M540 S1" or from the LCD
menu.
  // To have any effect, endstops must be enabled during SD
printing.
  //#define ABORT_ON_ENDSTOP_HIT_FEATURE_ENABLED

#endif // SDSUPPORT

// Some additional options are available for graphical displays:
#if ENABLED(DOGLCD)
  // A bigger font is available for edit items. Costs 3120 bytes of
PROGMEM.
  // Western only. Not available for Cyrillic, Kana, Turkish, Greek,
or Chinese.
  //#define USE_BIG_EDIT_FONT

  // A smaller font may be used on the Info Screen. Costs 2300 bytes
of PROGMEM.
  // Western only. Not available for Cyrillic, Kana, Turkish, Greek,
or Chinese.
  //#define USE_SMALL_INFOFONT

  // Enable this option and reduce the value to optimize screen
updates.
  // The normal delay is 10µs. Use the lowest value that still gives
```

```
a reliable display.
  //#define DOGM_SPI_DELAY_US 5
#endif // DOGLCD

// @section safety

// The hardware watchdog should reset the microcontroller disabling
all outputs,
// in case the firmware gets stuck and doesn't do temperature
regulation.
#define USE_WATCHDOG

#if ENABLED(USE_WATCHDOG)
  // If you have a watchdog reboot in an ArduinoMega2560 then the
device will hang forever, as a watchdog reset will leave the
watchdog on.
  // The "WATCHDOG_RESET_MANUAL" goes around this by not using the
hardware reset.
  //  However, THIS FEATURE IS UNSAFE!, as it will only work if
interrupts are disabled. And the code could hang in an interrupt
routine with interrupts disabled.
  //#define WATCHDOG_RESET_MANUAL
#endif

// @section lcd

// Babystepping enables the user to control the axis in tiny
amounts, independently from the normal printing process
// it can e.g. be used to change z-positions in the print startup
phase in real-time
// does not respect endstops!
#define BABYSTEPPING
#if ENABLED(BABYSTEPPING)
  #define BABYSTEP_XY  //not only z, but also XY in the menu. more
clutter, more functions
                       //not implemented for deltabots!
  #define BABYSTEP_INVERT_Z false  //true for inverse movements in Z
  #define BABYSTEP_MULTIPLICATOR 1 //faster movements
#endif

//
// Ensure Smooth Moves
//
// Enable this option to prevent the machine from stuttering when
printing multiple short segments.
// This feature uses two strategies to eliminate stuttering:
//
// 1. During short segments a Graphical LCD update may take so much
time that the planner buffer gets
//    completely drained. When this happens pauses are introduced
between short segments, and print moves
//    will become jerky until a longer segment provides enough time
for the buffer to be filled again.
//    This jerkiness negatively affects print quality. The
```

```
ENSURE_SMOOTH_MOVES option addresses the issue
//    by pausing the LCD until there's enough time to safely update.
//
//    NOTE: This will cause the Info Screen to lag and controller
buttons may become unresponsive.
//          Enable ALWAYS_ALLOW_MENU to keep the controller
responsive.
//
// 2. No block is allowed to take less time than MIN_BLOCK_TIME.
That's the time it takes in the main
//    loop to add a new block to the buffer, check temperatures,
etc., including all blocked time due to
//    interrupts (without LCD update). By enforcing a minimum time-
per-move, the buffer is prevented from
//    draining.
//
#define ENSURE_SMOOTH_MOVES
#if ENABLED(ENSURE_SMOOTH_MOVES)
  //#define ALWAYS_ALLOW_MENU      // If enabled, the menu will
always be responsive.
                                   // WARNING: Menu navigation
during short moves may cause stuttering!
  #define LCD_UPDATE_THRESHOLD 135 // (ms) Minimum duration for the
current segment to allow an LCD update.
                                   // Default value is good for
graphical LCDs (e.g.,
REPRAP_DISCOUNT_FULL_GRAPHIC_SMART_CONTROLLER).
                                   // You may try to lower this
value until you printer starts stuttering again as if
ENSURE_SMOOTH_MOVES is disabled.
  #define MIN_BLOCK_TIME 4         // (ms) Minimum duration of a
single block. You shouldn't need to modify this.
#endif

// @section extruder

// extruder advance constant (s2/mm3)
//
// advance (steps) = STEPS_PER_CUBIC_MM_E * EXTRUDER_ADVANCE_K *
cubic mm per second ^ 2
//
// Hooke's law says:    force = k * distance
// Bernoulli's principle says:  v ^ 2 / 2 + g . h + pressure /
density = constant
// so: v ^ 2 is proportional to number of steps we advance the
extruder
//#define ADVANCE

#if ENABLED(ADVANCE)
  #define EXTRUDER_ADVANCE_K .0
  #define D_FILAMENT 2.85
#endif

/**
```

```
 * Implementation of linear pressure control
 *
 * Assumption: advance = k * (delta velocity)
 * K=0 means advance disabled.
 * To get a rough start value for calibration, measure your "free
filament length"
 * between the hobbed bolt and the nozzle (in cm). Use the formula
below that fits
 * your setup, where L is the "free filament length":
 *
 * Filament diameter           |   1.75mm  |    3.0mm   |
 * ----------------------------|-----------|------------|
 * Stiff filament (PLA)        | K=47*L/10 | K=139*L/10 |
 * Softer filament (ABS, nGen) | K=88*L/10 | K=260*L/10 |
 */
//#define LIN_ADVANCE

#if ENABLED(LIN_ADVANCE)
  #define LIN_ADVANCE_K 75
#endif

// @section leveling

// Default mesh area is an area with an inset margin on the print
area.
// Below are the macros that are used to define the borders for the
mesh area,
// made available here for specialized needs, ie dual extruder
setup.
#if ENABLED(MESH_BED_LEVELING)
  #define MESH_MIN_X (X_MIN_POS + MESH_INSET)
  #define MESH_MAX_X (X_MAX_POS - (MESH_INSET))
  #define MESH_MIN_Y (Y_MIN_POS + MESH_INSET)
  #define MESH_MAX_Y (Y_MAX_POS - (MESH_INSET))
#endif

// @section extras

// Arc interpretation settings:
#define ARC_SUPPORT  // Disabling this saves ~2738 bytes
#define MM_PER_ARC_SEGMENT 1
#define N_ARC_CORRECTION 25

// Support for G5 with XYZE destination and IJPQ offsets. Requires
~2666 bytes.
//#define BEZIER_CURVE_SUPPORT

// G38.2 and G38.3 Probe Target
//#define G38_PROBE_TARGET
#if ENABLED(G38_PROBE_TARGET)
  #define G38_MINIMUM_MOVE 0.0275 // minimum distance in mm that
will produce a move (determined using the print statement in
check_move)
#endif
```

```
// Moves (or segments) with fewer steps than this will be joined
with the next move
#define MIN_STEPS_PER_SEGMENT 6

// The minimum pulse width (in µs) for stepping a stepper.
// Set this if you find stepping unreliable, or if using a very fast
CPU.
#define MINIMUM_STEPPER_PULSE 0 // (µs) The smallest stepper pulse
allowed

// @section temperature

// Control heater 0 and heater 1 in parallel.
//#define HEATERS_PARALLEL

//
===========================================================================
=======
//=============================== Buffers
===============================
//
===========================================================================
=======

// @section hidden

// The number of linear motions that can be in the plan at any give
time.
// THE BLOCK_BUFFER_SIZE NEEDS TO BE A POWER OF 2, i.g. 8,16,32
because shifts and ors are used to do the ring-buffering.
#if ENABLED(SDSUPPORT)
  #define BLOCK_BUFFER_SIZE 16 // SD,LCD,Buttons take more memory,
block buffer needs to be smaller
#else
  #define BLOCK_BUFFER_SIZE 16 // maximize block buffer
#endif

// @section serial

// The ASCII buffer for serial input
#define MAX_CMD_SIZE 96
#define BUFSIZE 4

// Transfer Buffer Size
// To save 386 bytes of PROGMEM (and TX_BUFFER_SIZE+3 bytes of RAM)
set to 0.
// To buffer a simple "ok" you need 4 bytes.
// For ADVANCED_OK (M105) you need 32 bytes.
// For debug-echo: 128 bytes for the optimal speed.
// Other output doesn't need to be that speedy.
// :[0, 2, 4, 8, 16, 32, 64, 128, 256]
#define TX_BUFFER_SIZE 0
```

```
// Enable an emergency-command parser to intercept certain commands
as they
// enter the serial receive buffer, so they cannot be blocked.
// Currently handles M108, M112, M410
// Does not work on boards using AT90USB (USBCON) processors!
//#define EMERGENCY_PARSER

// Bad Serial-connections can miss a received command by sending an
'ok'
// Therefore some clients abort after 30 seconds in a timeout.
// Some other clients start sending commands while receiving a
'wait'.
// This "wait" is only sent when the buffer is empty. 1 second is a
good value here.
//#define NO_TIMEOUTS 1000 // Milliseconds

// Some clients will have this feature soon. This could make the
NO_TIMEOUTS unnecessary.
//#define ADVANCED_OK

// @section fwretract

// Firmware based and LCD controlled retract
// M207 and M208 can be used to define parameters for the
retraction.
// The retraction can be called by the slicer using G10 and G11
// until then, intended retractions can be detected by moves that
only extrude and the direction.
// the moves are than replaced by the firmware controlled ones.

//#define FWRETRACT  //ONLY PARTIALLY TESTED
#if ENABLED(FWRETRACT)
  #define MIN_RETRACT 0.1                //minimum extruded mm to
accept a automatic gcode retraction attempt
  #define RETRACT_LENGTH 3               //default retract length
(positive mm)
  #define RETRACT_LENGTH_SWAP 13         //default swap retract
length (positive mm), for extruder change
  #define RETRACT_FEEDRATE 45            //default feedrate for
retracting (mm/s)
  #define RETRACT_ZLIFT 0                //default retract Z-lift
  #define RETRACT_RECOVER_LENGTH 0       //default additional
recover length (mm, added to retract length when recovering)
  #define RETRACT_RECOVER_LENGTH_SWAP 0  //default additional swap
recover length (mm, added to retract length when recovering from
extruder change)
  #define RETRACT_RECOVER_FEEDRATE 8     //default feedrate for
recovering from retraction (mm/s)
#endif

// Add support for experimental filament exchange support M600;
requires display
#if ENABLED(ULTIPANEL)
  // #define FILAMENT_CHANGE_FEATURE            // Enable filament
```

```
exchange menu and M600 g-code (used for runout sensor too)
  #if ENABLED(FILAMENT_CHANGE_FEATURE)
    #define FILAMENT_CHANGE_X_POS 3            // X position of
hotend
    #define FILAMENT_CHANGE_Y_POS 3            // Y position of
hotend
    #define FILAMENT_CHANGE_Z_ADD 10           // Z addition of
hotend (lift)
    #define FILAMENT_CHANGE_XY_FEEDRATE 100    // X and Y axes
feedrate in mm/s (also used for delta printers Z axis)
    #define FILAMENT_CHANGE_Z_FEEDRATE 5       // Z axis feedrate
in mm/s (not used for delta printers)
    #define FILAMENT_CHANGE_RETRACT_LENGTH 2   // Initial retract
in mm
                                               // It is a short
retract used immediately after print interrupt before move to
filament exchange position
    #define FILAMENT_CHANGE_RETRACT_FEEDRATE 60 // Initial retract
feedrate in mm/s
    #define FILAMENT_CHANGE_UNLOAD_LENGTH 100  // Unload filament
length from hotend in mm
                                               // Longer length for
bowden printers to unload filament from whole bowden tube,
                                               // shorter lenght
for printers without bowden to unload filament from extruder only,
                                               // 0 to disable
unloading for manual unloading
    #define FILAMENT_CHANGE_UNLOAD_FEEDRATE 10  // Unload filament
feedrate in mm/s - filament unloading can be fast
    #define FILAMENT_CHANGE_LOAD_LENGTH 0      // Load filament
length over hotend in mm
                                               // Longer length for
bowden printers to fast load filament into whole bowden tube over
the hotend,
                                               // Short or zero
length for printers without bowden where loading is not used
    #define FILAMENT_CHANGE_LOAD_FEEDRATE 10   // Load filament
feedrate in mm/s - filament loading into the bowden tube can be fast
    #define FILAMENT_CHANGE_EXTRUDE_LENGTH 50  // Extrude filament
length in mm after filament is load over the hotend,
                                               // 0 to disable for
manual extrusion
                                               // Filament can be
extruded repeatedly from the filament exchange menu to fill the
hotend,
                                               // or until
outcoming filament color is not clear for filament color change
    #define FILAMENT_CHANGE_EXTRUDE_FEEDRATE 3  // Extrude filament
feedrate in mm/s - must be slower than load feedrate
  #endif
#endif

/
**********************************************************************
```

```
**********\
 * enable this section if you have TMC26X motor drivers.
 * you need to import the TMC26XStepper library into the Arduino IDE
for this

************************************************************************
**********/

// @section tmc

//#define HAVE_TMCDRIVER
#if ENABLED(HAVE_TMCDRIVER)

  //#define X_IS_TMC
  //#define X2_IS_TMC
  //#define Y_IS_TMC
  //#define Y2_IS_TMC
  //#define Z_IS_TMC
  //#define Z2_IS_TMC
  //#define E0_IS_TMC
  //#define E1_IS_TMC
  //#define E2_IS_TMC
  //#define E3_IS_TMC

  #define X_MAX_CURRENT     1000 // in mA
  #define X_SENSE_RESISTOR    91 // in mOhms
  #define X_MICROSTEPS        16 // number of microsteps

  #define X2_MAX_CURRENT    1000
  #define X2_SENSE_RESISTOR   91
  #define X2_MICROSTEPS       16

  #define Y_MAX_CURRENT     1000
  #define Y_SENSE_RESISTOR    91
  #define Y_MICROSTEPS        16

  #define Y2_MAX_CURRENT    1000
  #define Y2_SENSE_RESISTOR   91
  #define Y2_MICROSTEPS       16

  #define Z_MAX_CURRENT     1000
  #define Z_SENSE_RESISTOR    91
  #define Z_MICROSTEPS        16

  #define Z2_MAX_CURRENT    1000
  #define Z2_SENSE_RESISTOR   91
  #define Z2_MICROSTEPS       16

  #define E0_MAX_CURRENT    1000
  #define E0_SENSE_RESISTOR   91
  #define E0_MICROSTEPS       16

  #define E1_MAX_CURRENT    1000
  #define E1_SENSE_RESISTOR   91
```

```
  #define E1_MICROSTEPS       16

  #define E2_MAX_CURRENT    1000
  #define E2_SENSE_RESISTOR   91
  #define E2_MICROSTEPS       16

  #define E3_MAX_CURRENT    1000
  #define E3_SENSE_RESISTOR   91
  #define E3_MICROSTEPS       16

#endif

// @section TMC2130


/**
 * Enable this for SilentStepStick Trinamic TMC2130 SPI-configurable
stepper drivers.
 *
 * To use TMC2130 drivers in SPI mode, you'll also need the TMC2130
Arduino library
 * (https://github.com/makertum/Trinamic_TMC2130).
 *
 * To use TMC2130 stepper drivers in SPI mode connect your SPI2130
pins to
 * the hardware SPI interface on your board and define the required
CS pins
 * in your `pins_MYBOARD.h` file. (e.g., RAMPS 1.4 uses AUX3 pins
`X_CS_PIN 53`, `Y_CS_PIN 49`, etc.).
 */

//#define HAVE_TMC2130DRIVER

#if ENABLED(HAVE_TMC2130DRIVER)

  //#define TMC2130_ADVANCED_CONFIGURATION

  // CHOOSE YOUR MOTORS HERE, THIS IS MANDATORY
  //#define X_IS_TMC2130
  //#define X2_IS_TMC2130
  //#define Y_IS_TMC2130
  //#define Y2_IS_TMC2130
  //#define Z_IS_TMC2130
  //#define Z2_IS_TMC2130
  //#define E0_IS_TMC2130
  //#define E1_IS_TMC2130
  //#define E2_IS_TMC2130
  //#define E3_IS_TMC2130

  #if ENABLED(TMC2130_ADVANCED_CONFIGURATION)

    // If you've enabled TMC2130_ADVANCED_CONFIGURATION, define
global settings below.
    // Enabled settings will be automatically applied to all axes
```

```
specified above.
    //
    // Please read the TMC2130 datasheet:
    // http://www.trinamic.com/_articles/products/integrated-
circuits/tmc2130/_datasheet/TMC2130_datasheet.pdf
    // All settings here have the same (sometimes cryptic) names as
in the datasheet.
    //
    // The following, uncommented settings are only suggestion.

    /* GENERAL CONFIGURATION */

    //#define GLOBAL_EN_PWM_MODE        0
    #define GLOBAL_I_SCALE_ANALOG     1 // [0,1] 0: Normal, 1: AIN
    //#define GLOBAL_INTERNAL_RSENSE    0 // [0,1] 0: Normal, 1:
Internal
    #define GLOBAL_EN_PWM_MODE        0 // [0,1] 0: Normal, 1:
stealthChop with velocity threshold
    //#define GLOBAL_ENC_COMMUTATION    0 // [0,1]
    #define GLOBAL_SHAFT              0 // [0,1] 0: normal, 1:
invert
    //#define GLOBAL_DIAG0_ERROR        0 // [0,1]
    //#define GLOBAL_DIAG0_OTPW         0 // [0,1]
    //#define GLOBAL_DIAG0_STALL        0 // [0,1]
    //#define GLOBAL_DIAG1_STALL        0 // [0,1]
    //#define GLOBAL_DIAG1_INDEX        0 // [0,1]
    //#define GLOBAL_DIAG1_ONSTATE      0 // [0,1]
    //#define GLOBAL_DIAG1_ONSTATE      0 // [0,1]
    //#define GLOBAL_DIAG0_INT_PUSHPULL 0 // [0,1]
    //#define GLOBAL_DIAG1_INT_PUSHPULL 0 // [0,1]
    //#define GLOBAL_SMALL_HYSTERESIS   0 // [0,1]
    //#define GLOBAL_STOP_ENABLE        0 // [0,1]
    //#define GLOBAL_DIRECT_MODE        0 // [0,1]

    /* VELOCITY-DEPENDENT DRIVE FEATURES */

    #define GLOBAL_IHOLD             22 // [0-31] 0: min, 31: max
    #define GLOBAL_IRUN              31 // [0-31] 0: min, 31: max
    #define GLOBAL_IHOLDDELAY        15 // [0-15] 0: min, 15: about
4 seconds
    //#define GLOBAL_TPOWERDOWN         0 // [0-255] 0: min, 255:
about 4 seconds
    //#define GLOBAL_TPWMTHRS           0 // [0-1048576] e.g. 20
corresponds with 2000 steps/s
    //#define GLOBAL_TCOOLTHRS          0 // [0-1048576] e.g. 20
corresponds with 2000 steps/s
    #define GLOBAL_THIGH             10 // [0-1048576] e.g. 20
corresponds with 2000 steps/s

    /* SPI MODE CONFIGURATION */

    //#define GLOBAL_XDIRECT            0

    /* DCSTEP MINIMUM VELOCITY */
```

```
    //#define GLOBAL_VDCMIN             0

    /* MOTOR DRIVER CONFIGURATION*/

    //#define GLOBAL_DEDGE              0
    //#define GLOBAL_DISS2G             0
    #define GLOBAL_INTPOL             1 // 0: off 1: 256 microstep
interpolation
    #define GLOBAL_MRES              16 // number of microsteps
    #define GLOBAL_SYNC               1 // [0-15]
    #define GLOBAL_VHIGHCHM           1 // [0,1] 0: normal, 1: high
velocity stepper mode
    #define GLOBAL_VHIGHFS            0 // [0,1] 0: normal, 1:
switch to full steps for high velocities
    // #define GLOBAL_VSENSE            0 // [0,1] 0: normal, 1:
high sensitivity (not recommended)
    #define GLOBAL_TBL                1 // 0-3: set comparator blank
time to 16, 24, 36 or 54 clocks, 1 or 2 is recommended
    #define GLOBAL_CHM                0 // [0,1] 0: spreadCycle, 1:
Constant off time with fast decay time.
    //#define GLOBAL_RNDTF              0
    //#define GLOBAL_DISFDCC            0
    //#define GLOBAL_FD                 0
    //#define GLOBAL_HEND               0
    //#define GLOBAL_HSTRT              0
    #define GLOBAL_TOFF              10 // 0: driver disable, 1: use
only with TBL>2, 2-15: off time setting during slow decay phase

    //#define GLOBAL_SFILT              0
    //#define GLOBAL_SGT                0
    //#define GLOBAL_SEIMIN             0
    //#define GLOBAL_SEDN               0
    //#define GLOBAL_SEMAX              0
    //#define GLOBAL_SEUP               0
    //#define GLOBAL_SEMIN              0

    //#define GLOBAL_DC_TIME            0
    //#define GLOBAL_DC_SG              0

    //#define GLOBAL_FREEWHEEL          0
    //#define GLOBAL_PWM_SYMMETRIC      0
    //#define GLOBAL_PWM_AUTOSCALE      0
    //#define GLOBAL_PWM_FREQ           0
    //#define GLOBAL_PWM_GRAD           0
    //#define GLOBAL_PWM_AMPL           0

    //#define GLOBAL_ENCM_CTRL          0

  #else

    #define X_IHOLD          31 // [0-31] 0: min, 31: max
    #define X_IRUN           31 // [0-31] 0: min, 31: max
    #define X_IHOLDDELAY     15 // [0-15] 0: min, 15: about 4
```

```
seconds
    #define X_I_SCALE_ANALOG  1 // 0: Normal, 1: AIN
    #define X_MRES           16 // number of microsteps
    #define X_TBL             1 // 0-3: set comparator blank time to
16, 24, 36 or 54 clocks, 1 or 2 is recommended
    #define X_TOFF            8 // 0: driver disable, 1: use only
with TBL>2, 2-15: off time setting during slow decay phase

    #define X2_IHOLD         31
    #define X2_IRUN          31
    #define X2_IHOLDDELAY    15
    #define X2_I_SCALE_ANALOG 1
    #define X2_MRES          16
    #define X2_TBL            1
    #define X2_TOFF           8

    #define Y_IHOLD          31
    #define Y_IRUN           31
    #define Y_IHOLDDELAY     15
    #define Y_I_SCALE_ANALOG  1
    #define Y_MRES           16
    #define Y_TBL             1
    #define Y_TOFF            8

    #define Y2_IHOLD         31
    #define Y2_IRUN          31
    #define Y2_IHOLDDELAY    15
    #define Y2_I_SCALE_ANALOG 1
    #define Y2_MRES          16
    #define Y2_TBL            1
    #define Y2_TOFF           8

    #define Z_IHOLD          31
    #define Z_IRUN           31
    #define Z_IHOLDDELAY     15
    #define Z_I_SCALE_ANALOG  1
    #define Z_MRES           16
    #define Z_TBL             1
    #define Z_TOFF            8

    #define Z2_IHOLD         31
    #define Z2_IRUN          31
    #define Z2_IHOLDDELAY    15
    #define Z2_I_SCALE_ANALOG 1
    #define Z2_MRES          16
    #define Z2_TBL            1
    #define Z2_TOFF           8

    #define E0_IHOLD         31
    #define E0_IRUN          31
    #define E0_IHOLDDELAY    15
    #define E0_I_SCALE_ANALOG 1
    #define E0_MRES          16
    #define E0_TBL            1
```

```
    #define E0_TOFF           8

    #define E1_IHOLD         31
    #define E1_IRUN          31
    #define E1_IHOLDDELAY    15
    #define E1_I_SCALE_ANALOG 1
    #define E1_MRES          16
    #define E1_TBL            1
    #define E1_TOFF           8

    #define E2_IHOLD         31
    #define E2_IRUN          31
    #define E2_IHOLDDELAY    15
    #define E2_I_SCALE_ANALOG 1
    #define E2_MRES          16
    #define E2_TBL            1
    #define E2_TOFF           8

    #define E3_IHOLD         31
    #define E3_IRUN          31
    #define E3_IHOLDDELAY    15
    #define E3_I_SCALE_ANALOG 1
    #define E3_MRES          16
    #define E3_TBL            1
    #define E3_TOFF           8

  #endif // TMC2130_ADVANCED_CONFIGURATION

#endif // HAVE_TMC2130DRIVER

// @section L6470

/**
 * Enable this section if you have L6470 motor drivers.
 * You need to import the L6470 library into the Arduino IDE for
this.
 * (https://github.com/ameyer/Arduino-L6470)
 */

//#define HAVE_L6470DRIVER
#if ENABLED(HAVE_L6470DRIVER)

  //#define X_IS_L6470
  //#define X2_IS_L6470
  //#define Y_IS_L6470
  //#define Y2_IS_L6470
  //#define Z_IS_L6470
  //#define Z2_IS_L6470
  //#define E0_IS_L6470
  //#define E1_IS_L6470
  //#define E2_IS_L6470
  //#define E3_IS_L6470

  #define X_MICROSTEPS      16 // number of microsteps
```

```
  #define X_K_VAL           50 // 0 - 255, Higher values, are higher
power. Be careful not to go too high
  #define X_OVERCURRENT   2000 // maxc current in mA. If the current
goes over this value, the driver will switch off
  #define X_STALLCURRENT  1500 // current in mA where the driver
will detect a stall

  #define X2_MICROSTEPS     16
  #define X2_K_VAL          50
  #define X2_OVERCURRENT  2000
  #define X2_STALLCURRENT 1500

  #define Y_MICROSTEPS      16
  #define Y_K_VAL           50
  #define Y_OVERCURRENT   2000
  #define Y_STALLCURRENT  1500

  #define Y2_MICROSTEPS     16
  #define Y2_K_VAL          50
  #define Y2_OVERCURRENT  2000
  #define Y2_STALLCURRENT 1500

  #define Z_MICROSTEPS      16
  #define Z_K_VAL           50
  #define Z_OVERCURRENT   2000
  #define Z_STALLCURRENT  1500

  #define Z2_MICROSTEPS     16
  #define Z2_K_VAL          50
  #define Z2_OVERCURRENT  2000
  #define Z2_STALLCURRENT 1500

  #define E0_MICROSTEPS     16
  #define E0_K_VAL          50
  #define E0_OVERCURRENT  2000
  #define E0_STALLCURRENT 1500

  #define E1_MICROSTEPS     16
  #define E1_K_VAL          50
  #define E1_OVERCURRENT  2000
  #define E1_STALLCURRENT 1500

  #define E2_MICROSTEPS     16
  #define E2_K_VAL          50
  #define E2_OVERCURRENT  2000
  #define E2_STALLCURRENT 1500

  #define E3_MICROSTEPS     16
  #define E3_K_VAL          50
  #define E3_OVERCURRENT  2000
  #define E3_STALLCURRENT 1500

#endif
```

```
/**
 * TWI/I2C BUS
 *
 * This feature is an EXPERIMENTAL feature so it shall not be used
on production
 * machines. Enabling this will allow you to send and receive I2C
data from slave
 * devices on the bus.
 *
 * ; Example #1
 * ; This macro send the string "Marlin" to the slave device with
address 0x63 (99)
 * ; It uses multiple M260 commands with one B<base 10> arg
 * M260 A99  ; Target slave address
 * M260 B77  ; M
 * M260 B97  ; a
 * M260 B114 ; r
 * M260 B108 ; l
 * M260 B105 ; i
 * M260 B110 ; n
 * M260 S1   ; Send the current buffer
 *
 * ; Example #2
 * ; Request 6 bytes from slave device with address 0x63 (99)
 * M261 A99 B5
 *
 * ; Example #3
 * ; Example serial output of a M261 request
 * echo:i2c-reply: from:99 bytes:5 data:hello
 */

// @section i2cbus

//#define EXPERIMENTAL_I2CBUS
#define I2C_SLAVE_ADDRESS  0 // Set a value from 8 to 127 to act as
a slave

/**
 * Add M43 command for pins info and testing
 */
//#define PINS_DEBUGGING

/**
 * Auto-report temperatures with M155 S<seconds>
 */
//#define AUTO_REPORT_TEMPERATURES

/**
 * Include capabilities in M115 output
 */
//#define EXTENDED_CAPABILITIES_REPORT

#endif // CONFIGURATION_ADV_H
```